REMARKS

This Amendment responds to the Office Action dated July 26, 2005 in which the Examiner objected to claims 7, 10 and 20 and rejected claims 1-26 under 35 U.S.C. §103.

As indicated above, claim 7 has been amended to further limit the subject matter of the previous claim. Additionally, claims 10 and 20 have been canceled without prejudice. Therefore, applicants respectfully request the Examiner withdraws the objection to the claims.

Claim 1 and 11 claim a data processing system comprising a plurality of processors and a memory. The plurality of processors executes a series of different types of processing functions on data to be processed in a prescribed order. Each processor executes a processing function different from one another. The data to be processed is image data that consists of a plurality of pixel data. The memory stores the data to be processed in association with state information representing the processing to be performed next for each pixel data to be processed. Processing functions are asynchronously executed on the data to be processed by the plurality of processors. One processing is executed on each pixel data by one processor at a time. The plurality of processors shares the memory.

Through the structure of the claimed invention having a plurality of processors execute a series of different types of processing functions on data to be processed on each pixel by one processor at a time as claimed in claims 1 and 11, the claimed invention provides a data processing system capable of processing data at high speed while allowing memory capacity to be reduced. The prior art does not show, teach or suggest the invention as claimed in claims 1 and 11.

Claim 25 claims a data processing device and claim 26 claims an image processing device. The device is comprises first and second (image) processors and a memory. The first (image) processor executes first (image) processing on data. The second (image) processor executes second (image) processing on the data that is subjected to the first processing. The memory stores the (image) data in association with state information to represent the processing state of the (image) data. The first and second (image) processors are asynchronously executed on the (image) data by the first and second (image) processors. The first and second (image) processors share the memory.

Through the structure of the claimed invention having a first processor execute first processing on data and a second processor executing second processing on the data that is subjected to the first processing, as claimed in claims 25 and 26, the claimed invention provides a data processing system capable of processing data at high speed while allowing memory capacity to be reduced. The prior art does not show, teach or suggest the invention as claimed in claims 25 and 26.

Claims 25 and 26 were rejected under 35 U.S.C. §103 as being unpatentable over *Orimo et al* (U.S. Patent No. 5,630,135) in view of *Charles et al* (U.S. Patent No. 5,790,842).

Applicants respectfully traverse the Examiner's rejection of the claims under 35 U.S.C. §103. The claims have been reviewed in light of the Office Action, and for reasons which will be set forth below, applicants respectfully request the Examiner withdraws the rejection to the claims and allows the claims to issue.

Orimo et al appears to disclose "multiple-version programs," which means a plurality of programs for performing the same function but having different program structures. The execution results of the programs may be either the same or not the same. (col. 1, lines 18-22) In a distributed processing system having a plurality of processors connected through a network, at least two first processors execute multiple-version programs which perform the same function, and messages which contain data output as execution results of the programs and attribute information indicating the versions of the executed programs are sent from the first processors to the network. The messages containing the results of processing by the multiple-version programs sent from the first processors are received by a second processor, which selects one message from the received messages based on the attribute information contained in the received messages and executes a program in the second processor by using the data contained in the selected message. In one embodiment, the second processor selects those messages sent from the first processors which were received in a predetermined permissible time period, as candidates for selection. Of the candidate messages, the message sent by the program having the highest priority version is determined to select one message. The multiple-version programs executed by the first processors may include programs having different calculation precision or programs having different developed time frames. (col. 2, lines 2-25) As shown therein, multiple-execution system 100 for the multiple-version programs comprises processors 11, 12, 13 and 14 connected to any type of network. Each of the processors 11, 12, 13 and 14 stores an application program in an associated internal memory and executes the application program to conduct various processings. Each application program has a

name assigned to correspond to a function and attribute, including version information and is managed thereby. Where the multiple-version programs performing the same function are not present, the attribute thereof is "Null". FIG. 2 is a diagram of a main part of a format of a message flowing over the network 1. Data transmitted among the processors by the message is stored in a data field 205. A CC field 201 stores a content code indicating the content of the data stored in the data field 205. The processors 11-14 connected to the network determine whether to read in the message flowing over the network 1 or not based on the content code of the CC field 201. A name field 203 stores information for identifying an application program generated by the message. In the present embodiment, a name of the application program is used as the information to be stored in the name field 203. An ATR field 204 stores an attribute of the application program generated by the message. (col. 3, line 57 through col. 4, line 16) In FIG. 10, application programs 71a, 71b and 71c of different versions are arranged in the processors 11, 12 and 13, respectively. The application programs 71a, 71b and 71c are those of different versions which perform the same function, and the names thereof are "X" and the attributes are "1", "2" and "3", respectively. The application program 71a performs a simulation at a high precision based on a complex logic and has a long calculation time. On the other hand, the application program 71c performs a simulation at a low precision based on a simple logic and a calculation time thereof is short. The precision and the calculation time of the application program 71b are in between those of the application program 71a and the application program 71c. Application programs 72 and 73 are arranged on the processor 14 and a terminal 1401 is connected to the processor 14 as interface means with a user. The application

program 72 requests the simulation based on a request inputted from the terminal 1401. The application program 73 receives the result of the simulation to output it to the terminal 1401. When the processor 14 receives the request from the terminal 1401, it executes the processing by the application program 72 and outputs a message 720 requesting the simulation to the network 1. When the processors 11, 12 and 13 receives the message 720, they start the execution of the application programs 71a, 71b and 71c, respectively. The execution result of the application program 71c is first outputted to the network 1 as a message 710c. Thereafter, the execution result of the application program 71b is outputted to the network 1 as a message 710b, and the execution result of the application program 71a is finally outputted to the network as a message 710a. The processor 14 receives the messages 710c, 710b and 710a in this sequence and executes the processing by the application program 73 each time, it receives the application program and displays the content of the received message on the terminal 1401. In this manner, the result of the low precision simulation is first displayed on the terminal 1401, and then the result of the middle precision simulation is displayed. Finally, the result of the high precision simulation is displayed. Thus, the user can previously acquire a general result before he/she acquires the result of the high precision simulation. Since the user can grasp the relation between the results of gradually increasing precision, the user interface is improved. For example, assuming that the application programs 71a, 71b and 71c are image analysis programs of different analysis precisions, a coarse image is displayed on the terminal 1401 shortly after the request for process, and the image of higher precision are displayed as the time elapses. (col. 9, line 55 through col. 10, line 36) In accordance with the program execution

method and the processing system, the programs which perform the same function but have different execution logics or execution precisions are executed in parallel, and one of the execution results is selected based on the versions of the programs, the contents of the execution results and the output times. (col. 10, lines 50-56)

Thus, *Orimo et al* merely discloses in FIG. 10 a processor 14 which causes an application program 72 to request simulation based upon a request input from a terminal 1401 and when it receives the results of the simulation causes a program 73 to output the result to terminal 1401 (col. 2, lines 2-25, col. 10, lines 1-23). Thus, nothing in *Orimo et al* shows, teaches or suggests a first processor which executes processing on data and a second image processor which executes processing on data that was subjected to the first processing as claimed in claims 25 and 26. Rather, *Orimo et al* merely discloses outputting a simulation request when a request is input from a terminal, where the message contains data. However, *Orimo et al* does not show, teach or suggest that the data that is passed to the subsequent processors has been processed itself by processor 14. Therefore, the processors 11-13 of *Orimo et al* which receive the simulation request from processor 14 do not execute processing on data that was subjected to first processing. Rather, the data input to the processors 11-13 of *Orimo et al* is merely passed along in the simulation request.

Additionally, processors 11-13 of *Orimo et al* perform different versions of the same function (col. 9, lines 55-67). Thus, nothing in *Orimo et al* shows, teaches or suggests a second processor executing processing on data that is subjected to first processing as claimed in claims 25 and 26. Rather, processors 11-13 of *Orimo et al* perform the same function but have different execution logic.

Also, the Examiner stated that FIG. 1 of *Orimo et al* discloses first and second processors. However, *Orimo et al* merely discloses in FIG. 1 that the processors 11-14 perform multi-version programs, i.e. perform the same function. Thus nothing in *Orimo et al* shows, teaches or suggests a second processor executing processing on data subjected to first processing as claimed in claims 25 and 26.

Finally, processors 11-13 of *Orimo et al* are executed in parallel (col. 10, lines 50-56). Nothing in *Orimo et al* shows, teaches or suggests first and second processing are asynchronously executed as claimed in claims 25 and 26. Rather, *Orimo et al* clearly teaches away from the claimed invention since the processors 11-13 are executed in parallel.

Charles et al appears to disclose memory arbitration techniques which allow multiple processes to share a common memory device or devices. In the exemplary processing system of FIG. 1, the memory arbitration techniques permit a number of graphics, communication and other processes operating within ASIC processor 20 to share the DRAM 40. This memory arbitration eliminates the requirement for separate memory devices in multiple processing elements, and thus permits a more efficient and cost-effective processing system implementation. (col. 27, lines 54-63)

Thus, *Charles et al* merely discloses a shared memory device. Nothing in *Charles et al* shows, teaches or suggests first and second processors as claimed in claims 25 and 26.

Since nothing in *Orimo et al* or *Charles et al* shows, teaches or suggests a first processor executing first processing on data and a second processor executing second processing on the data that was subjected to the first processing which is asynchronously executed as claimed in claims 25 and 26, applicants respectfully

request the Examiner withdraws the rejection to claims 25 and 26 under 35 U.S.C. §103.

Claims 1-24 were rejected under 35 U.S.C. §103 as being unpatentable over *Orimo et al* in view of *FOLDOC* (http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=image) and *Charles et al*.

Applicants respectfully traverse the Examiner's rejection of the claims under 35 U.S.C. §103. The claims have been reviewed in light of the Office Action, and for reasons which will be set forth below, applicants respectfully request the Examiner withdraws the rejection to the claims and allows the claims to issue.

As discussed above, *Orimo et al* at column 2, lines 1-25, is directed to the embodiment shown in Figure 10 in which processor 14 does not process data but merely requests simulation and passes the data to processors 11-13. However, processors 11-13 of *Orimo et al* execute the same versions of the same program which perform the same function (col. 1, lines 18-20, col. 9, lines 57-59 and col. 10, lines 51-56). Nothing in *Orimo et al* shows, teaches or suggests that each processor executes a processing function different from one another as claimed in claims 1 and 11. Rather, *Orimo et al* merely discloses executing different versions of the same program (i.e., same processing functions).

Furthermore, *Orimo et al* merely discloses the processors 11-13 process the data in parallel (col. 8, lines 34-37, col. 10, lines 11-13 and 50-54). However, as claimed in claims 1 and 11, one processing is executed on each pixel by one processor at a time asynchronously. However, *Orimo et al* teaches away from the claimed invention and processes the data in parallel.

FOLDOC merely discloses a digital image as composed of pixels arranged in a rectangular array with a certain height and width. Nothing in *FOLDOC* shows, teaches or suggests each processor executing a processing function different from one another and that the processing functions are asynchronously executed as claimed in claims 1 and 11. Rather, *FOLDOC* merely discloses a digital image is composed of pixels arranged in a rectangular array.

As discussed above, *Charles et al* merely discloses a common memory. Nothing in *Charles et al* shows, teaches or suggests each processor executing a processing function different from one another and that the processing functions are asynchronously executed as claimed in claims 1 and 11.

Since nothing in *Orimo et al*, *FOLDOC* and *Charles et al* shows, teaches or suggests each processor executing a processing function different from one another and the processing functions are asynchronously executed as claimed in claims 1 and 11, applicants respectfully request the Examiner withdraws the rejection to claims 1 and 11 under 35 U.S.C. §103.

Claims 2-9, 12-19, and 21-24 depend from claims 1 and 11 and recite additional features. Applicants respectfully submit that claims 2-10 and 12-24 would not have been obvious within the meaning of 35 U.S.C. §103 over *Orimo et al*, *FOLDOC* and *Charles et al* at least for the reasons as set forth above. Therefore, applicants respectfully request the Examiner withdraws the rejection to claims 2-9, 12-19, and 21-24 under 35 U.S.C. §103.

Thus it now appears that the application is in condition for reconsideration and allowance. Reconsideration and allowance at an early date are respectfully requested.

If for any reason the Examiner feels that the application is not now in condition for allowance, it is respectfully requested that the Examiner contact, by telephone, the Applicants' undersigned attorney at the indicated telephone number to arrange for an interview to expedite the disposition of this case.

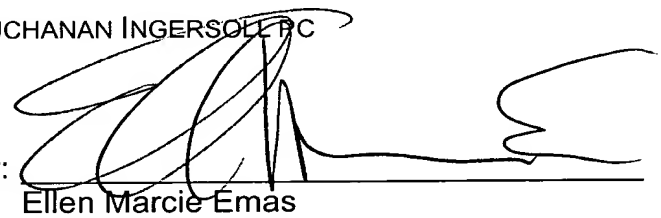
In the event that this paper is not timely filed within the current set shortened statutory period, Applicants respectfully petition for an appropriate extension of time. The fees for such extension of time may be charged to our Deposit Account No. 02-4800.

In the event that any additional fees are due with this paper, please charge our Deposit Account No. 02-4800.

Respectfully submitted,

BUCHANAN INGERSOLL PC

Date: October 25, 2005

By: 

Ellen Marcie Emas
Registration No. 32,131

P.O. Box 1404
Alexandria, Virginia 22313-1404
(703) 836-6620